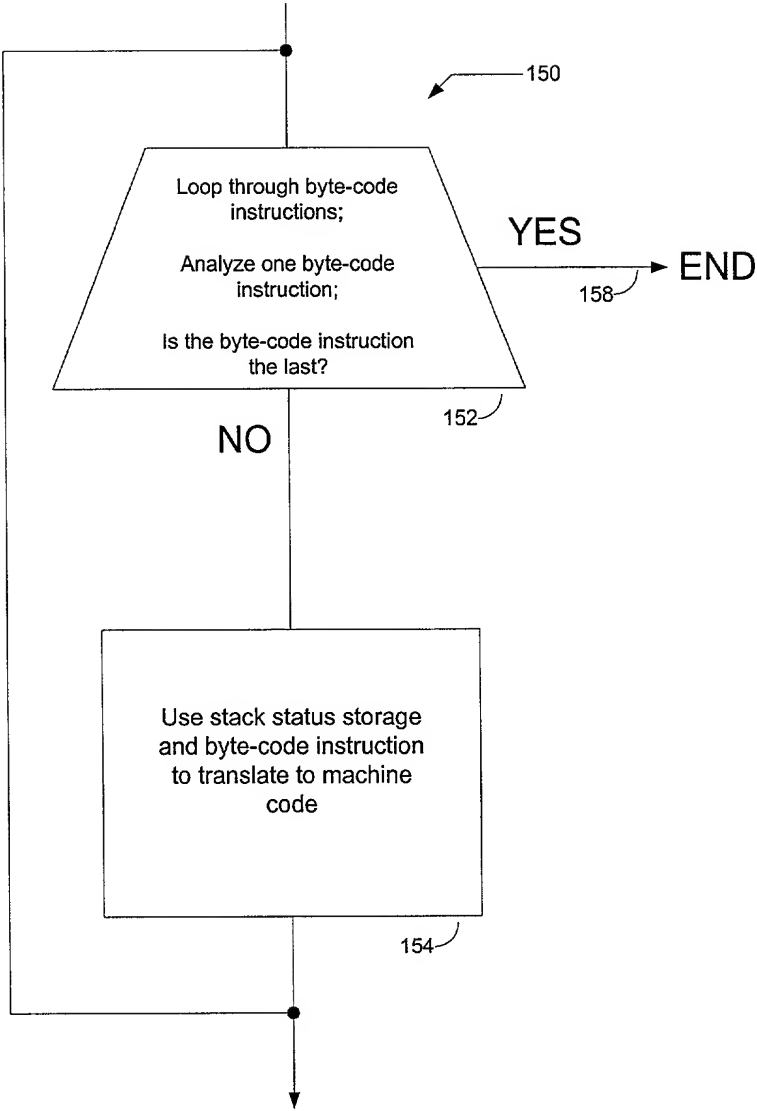


Traditional Byte-
Code Compilation

Pass 1

FIGURE 1A



Traditional Byte-Code Compilation

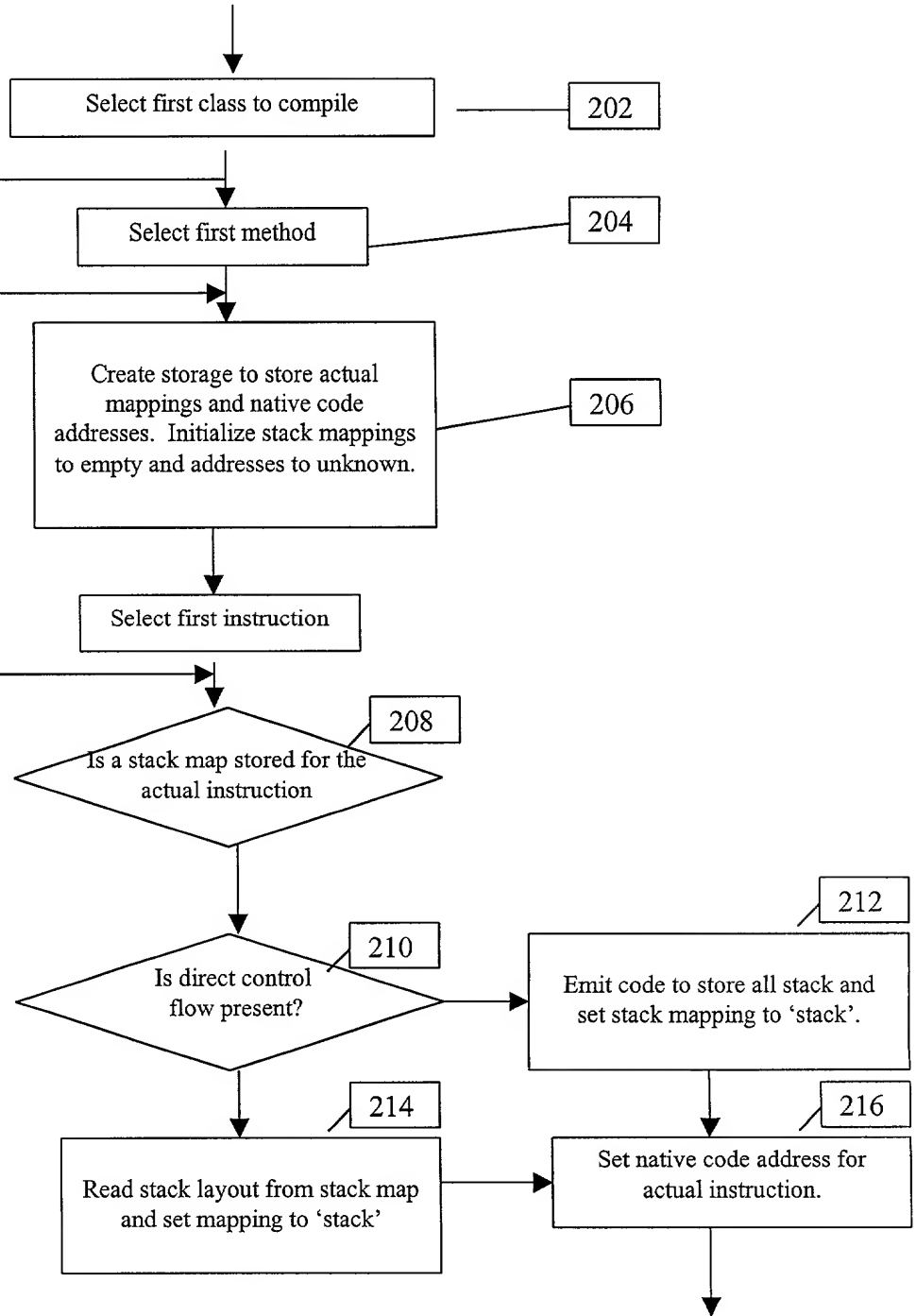
Pass 2

FIGURE 1B

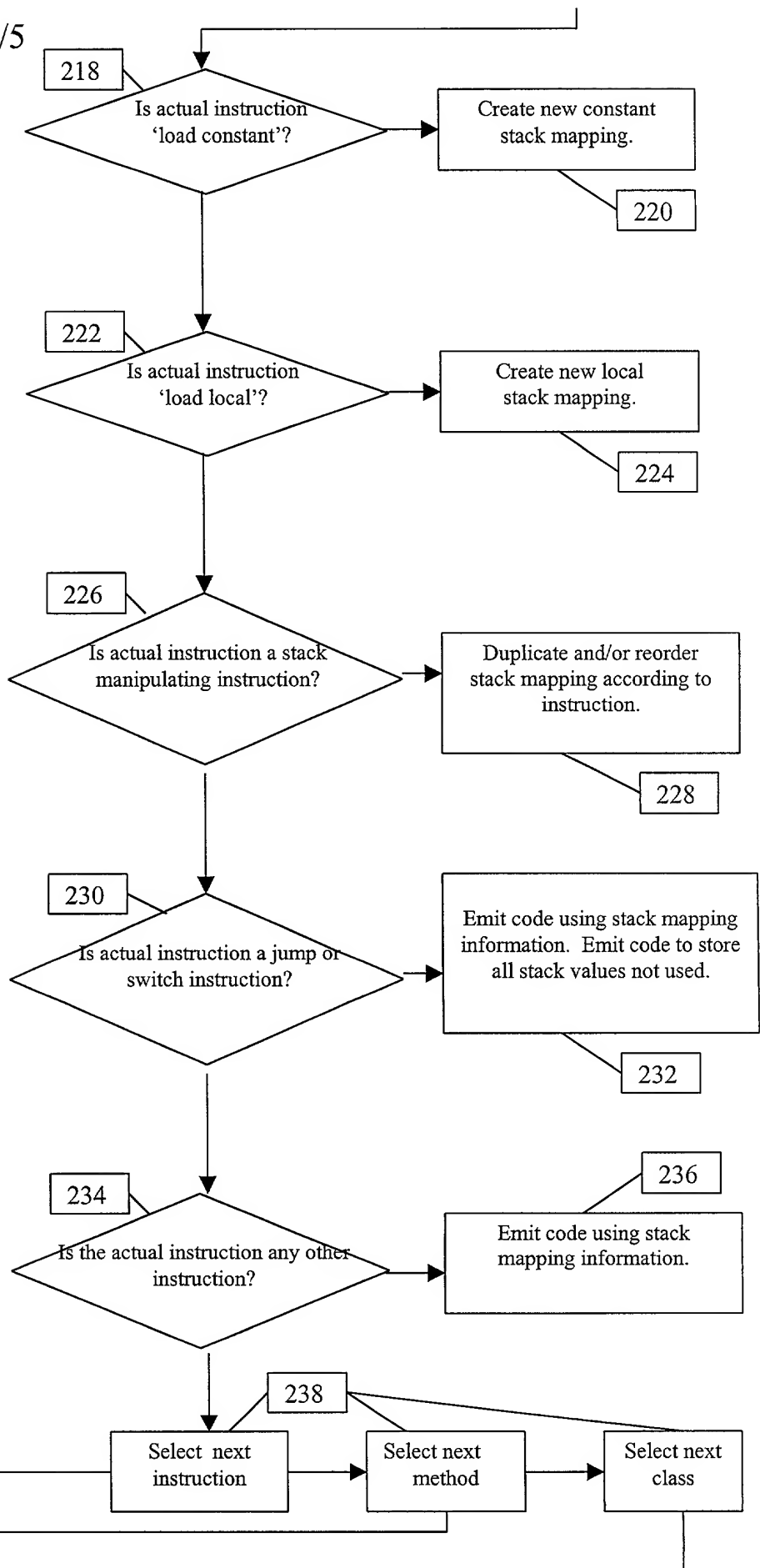
```

graph TD
    Start(( )) --> 202[Select first class to compile]
    202 --> 204[Select first method]
    204 --> 206[Create storage to store actual mappings and native code addresses. Initialize stack mappings to empty and addresses to unknown.]
    206 --> 208[Select first instruction]
    208 --> D1{Is a stack map stored for the actual instruction}
    D1 -- Yes --> D2{Is direct control flow present?}
    D1 -- No --> D2
    D2 -- Yes --> 212[Emit code to store all stack and set stack mapping to 'stack'.]
    D2 -- No --> 214[Read stack layout from stack map and set mapping to 'stack']
    212 --> 216[Set native code address for actual instruction.]
    214 --> 216
    216 --> End(( ))
    204 -- Loop --> 206
    204 -- Loop --> 208
    202 -- Loop --> 204

```



4/5



Arrays of Fixed Size

For each value on the bytecode stack	A field showing actual mapping to storage in target machine	constant
		local
		temporary
		stack
	A field containing additional information	constant value
		slot number
		register number
For each target of a jump or switch instruction	A field to store native code address	

Required Data Structures

FIGURE 3